Trial Division algorithm

Is n prime?

Idea: try to divide n by $2, 3, \ldots, \sqrt{n}$

If $k, m > \sqrt{n}$, then $km > n$.

Thus, if n is not prime, then it has a factor $< \sqrt{n}$.

Implementation 1:

```
isPrime[n_Integer] := Module[{},
  Do[
   If[Divisible[n, k], Return[False, Module]],
   {k, 2, Sqrt[n]}];
  Return[True]
  ]
```

makes Mathematica quit the module

If we get here, no divisor was found, so n is prime.

Implementation 2:

```
isPrime2[n_] := Module[ {foundDivisor = False, d = 2, m = Sqrt[n]},
  While[ ! foundDivisor && d ≤ m,
   If[Divisible[n, d], foundDivisor = True];
   d++
  ];

  ! foundDivisor (* return value *)
 ]
```

test divisor

NOT

AND

add 1 to d

Implementation 3:

```
isPrime3[n_Integer] := Module[{m = Sqrt[n]},
  Catch[
   Do[
    If[Divisible[n, d], Throw[False]],
    {d, 2, m}
   ];
   True
  ] (* the Module returns the "caught" value *)
]
```

transfers control to the enclosing Catch statement, which takes the value False

returned if no divisor is found

SIEVE OF ERATOSTHENES: how to implement this?

2 3 ~~4~~ 5 ~~6~~ 7 ~~8~~ ~~9~~ ~~10~~ 11 ~~12~~

13 ~~14~~ ~~15~~ ~~16~~ 17 ~~18~~ 19 ~~20~~ ~~21~~ ~~22~~ 23 = n

input: integer n (big!)

output: list of all primes up to n